UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/837,929	04/19/2001	William J. Walker	500007-A-01-US (Walker)	8223

2292 7590 11/08/2005

BIRCH STEWART KOLASCH & BIRCH
PO BOX 747
FALLS CHURCH, VA 22040-0747

EXAMINER	
RUTTEN, JAMES D	

ART UNIT	PAPER NUMBER
2192	

DATE MAILED: 11/08/2005

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary	Application No.	Applicant(s)	
	09/837,929	WALKER, WILLIAM J.	
	Examiner	Art Unit	
	J. Derek Rutten	2192	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *15 September 2005*.

2a)☒ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-13* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-13* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☐ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .
5) ☐ Notice of Informal Patent Application (PTO-152)
6) ☐ Other: _____.

DETAILED ACTION

Continued Examination Under 37 CFR 1.114

1. A request for continued examination under 37 CFR 1.114, including the fee set forth in 37 CFR 1.17(e), was filed in this application after final rejection. Since this application is eligible for continued examination under 37 CFR 1.114, and the fee set forth in 37 CFR 1.17(e) has been timely paid, the finality of the previous Office action has been withdrawn pursuant to 37 CFR 1.114. Applicant's submission filed on 15 September 2005 has been entered.

Declaration - 37 CFR 1.131

2. The declaration filed on 15 September 2005 under 37 CFR 1.131 has been considered but is ineffective to overcome the "McLaughlin-3" or "McLaughlin-4" references. Applicant has attempted to show conception of the invention prior to January 26, 2000, coupled with diligence at least from September 2000 (date of the McLaughlin-3 reference) to April 19, 2001 (date of filing).

The essential thing to be shown under 37 CFR 1.131 is priority of invention and this may be done by any satisfactory evidence of the fact. Any exhibit relied upon to provide such evidence should be specifically referred to in the declaration, **in terms of what it is relied upon to show**. A general allegation that the invention was completed prior to January 26 2000 is not sufficient. *Ex parte Saunders*, 1883 C.D. 23, 23 O.G. 1224 (Comm'r Pat. 1883). Similarly, a declaration by the inventor to the effect that his or her invention was conceived or reduced to practice prior to September 2000 (*McLaughlin-3*), without a statement of **facts demonstrating**

the correctness of this conclusion, is insufficient to satisfy 37 CFR 1.131. An accompanying exhibit need not support all claimed limitations, provided that any missing limitation is supported by the declaration itself. *Ex parte Ovshinsky*, 10 USPQ2d 1075 (Bd. Pat. App. & Inter. 1989). The declaration must **clearly explain which facts or data applicant is relying on** to show completion of his or her invention prior to September 2000. **Vague and general statements in broad terms** about what the exhibits describe along with a general assertion that the exhibits describe a reduction to practice "amounts essentially to mere pleading, unsupported by proof or a showing of facts" and, thus, does not satisfy the requirements of 37 CFR 1.131(b). *In re Borkowski*, 505 F.2d 713, 184 USPQ 29 (CCPA 1974). Applicant must give a clear explanation of the exhibits pointing out **exactly what facts are established and relied on** by applicant. 505 F.2d at 718-19, 184 USPQ at 33. See also *In re Harry,* 333 F.2d 920, 142 USPQ 164 (CCPA 1964) (Affidavit "asserts that facts exist but does not tell what they are or when they occurred.").

In the description of exhibit 1 on page 1 of the declaration filed on 15 September 2005, Applicant merely asserts that written description support for claims 1-13 appears on page 3 line 14 – page 7 line 15 of "A Mechanism for Converting Between Java Classes and XML" (THE DOCUMENT). However, the indication of a range of 5 pages in support of 13 claims is considered to be a vague and general statement in broad terms. Further, no clear explanation has been given regarding THE DOCUMENT in terms of the invention of claims 1-13. The declaration does not meet the requirements for providing exhibits establishing conception since it merely provides vague and general statements in broad terms and does not clearly explain which portions of THE DOCUMENT are being relied upon to show conception in support of the limitations of claims 1-13. If supported by an appropriate declaration or affidavit which clearly

explains which facts or data applicant is relying upon, THE DOCUMENT would appear to be largely sufficient to show conception since it shares much of the text of the originally filed specification. An accompanying exhibit need not support all claimed limitations, provided that any missing limitation is supported by the declaration itself. However, since a clear explanation pointing out what facts are relied upon has not been submitted, conception of the invention has not been sufficiently shown, and thus the declaration is ineffective to overcome the *McLaughlin-3* and *McLaughlin-4* references. See MPEP 715.07(I).

In the interest of compact prosecution, it is noted that due diligence has not been sufficiently established through a clear explanation of any periods of inactivity. Applicant has submitted numerous documents and has provided an identifying description of these documents in the declaration. However, there remain large periods of inactivity, especially the period from September 2000 (date of the McLaughlin-3 reference) to April 2001 (date of filing), which have not been clearly explained. Exhibits 6-8 (dated May 8, 2000 – August 3, 2000) appear to indicate that no work had been done on the application for six months: from the time it was first docketed on January 31, 2000 (presented in exhibit 5) until at least August 3, 2000 when it was transferred to AVAYA, Inc. Almost two months later, a status inquiry dated September 26, 2000 is provided by exhibit 10. Exhibit 14, dated October 20, 2000 (almost one month after the initial status request) simply provides a priority request. After this priority request, an unexplained period of almost four months passes before the date of exhibit 15 on February 8, 2001 when THE DOCUMENT was then forwarded to outside counsel for application preparation. From the date of exhibit 15, another two and a half months pass before the actual filing of the application. In total, there are at least fourteen months of apparent inactivity that are not clearly explained. It is

noted that over a year passes from the initial submission of THE DOCUMENT on January 26, 2000 (exhibit 2) until the application is transferred to outside counsel on February 8, 2001 (exhibit 15). Although applicant made 2 status inquiries (exhibits 6 and 10), no activity is alleged to have occurred during this time and so there is no activity to rely on for diligence. Further, a clear explanation of any periods of inactivity should be presented. Since conception of the invention has not been clearly established, due diligence has not been considered. Thus, these observations are merely exemplary and are not comprehensive. In determining the sufficiency of a 37 CFR 1.131 affidavit or declaration, diligence need not be considered unless conception of the invention prior to the effective date is clearly established, since diligence comes into question only after prior conception is established. *Ex parte Kantor*, 177 USPQ 455 (Bd. App. 1958). Also see MPEP 715.07(a).

The examiner is required to make a legal determination based upon the facts submitted in the declaration. While the associated documents may or may not provide sufficient support for conception of the invention, the decision is made based upon the statements found in the declaration. Without a clear explanation of which facts or data applicant is relying on to show completion of the claimed invention prior to a particular date, combined with a clear explanation of any periods of activity and/or inactivity, the declaration is not sufficient to overcome the reference.

Claim Rejections

3. The text of the following rejections of claims 1-13 under 35 USC 102 and 103(a) is copied from the from the previous Office Action mailed on 17 June 2005, and is included for convenience.

Claim Rejections - 35 USC § 102

4. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

> A person shall be entitled to a patent unless –
>
> (a) the invention was known or used by others in this country, or patented or described in a printed publication in this or a foreign country, before the invention thereof by the applicant for a patent.

5. Claim 2 is rejected under 35 U.S.C. 102(a) as being anticipated by prior art of record "Data Binding from XML to Java, Part 3" by McLaughlin, published September 2000 (hereinafter referred to as "McLaughlin-3").

The text of the rejection of claim 2 below is reproduced for convenience from the previous Office action filed 3 January 2005.

In regard to claim 2, McLaughlin-3 discloses:

instantiating an object of a desired object-oriented programming language class (See page 3 paragraph 2: "Once a new instance of the appropriate class is created...");

in the case of said instantiated object, implementing a predefined interface (See page 3 paragraph 2: "Once a new instance of the appropriate class is created, the mutator methods are called with the values supplied in the XML document." The mutator

methods are part of a predefined interface for setting attribute values. Listing 3 on page 3 shows the corresponding implementation of a mutator method. Mutator methods are commonly used for setting attribute values.),

iteratively processing each object included within said instantiated object (See page 4 paragraph 4: "Once all the attributes are read and assigned to the created Java instance, you need to take each nested element and perform the unmarshalling again.")

according to the steps of:

retrieving field descriptors associated with said object being processed (See page 4 paragraph 3: "Finally, the nested objects are passed to **accessor methods**, and the top-level object, which is populated with both member variable values and object **references**, is **returned** to the calling program." Accessor methods are commonly used to retrieve attribute values including field descriptors.);

creating an object of a specified desired object-oriented programming language type for each XML element corresponding to a field descriptor (See McLaughlin-3 page 3 paragraph 2: "Once a new instance of the appropriate class is created, the mutator methods are called with the values supplied in the XML document."; also page 4 paragraph 4: "Once all the attributes are read and assigned to the created Java instance, you need to take each nested element and perform the unmarshalling again."); *and*

storing the created object in the currently processed object (See page 4 paragraph 4: "Once all the attributes are read and assigned to the created Java instance...").

Claim Rejections - 35 USC § 103

6. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

7. Claims 1 and 3-9 are rejected under 35 U.S.C. 103(a) as being unpatentable over prior art of record "Data Binding from XML to Java, Part 4" by McLaughlin published on October 1, 2000 (hereinafter referred to as "McLaughlin-4") in view of McLaughlin-3.

The text of the rejections of claims 1 and 3-9 below is reproduced for convenience from the previous Office action filed 19 May 2004, in light of the amendment filed 10 September 2004.

As per claim 1, McLaughlin-4 discloses:

loading the named object-oriented programming language class (See page 2 paragraph 7: "It takes an object and should return an XML element representation of that object." An object is inherently loaded from an object-oriented programming language class.);

determining if the loaded object-oriented programming language class implements a predefined interface (See page 2 paragraph 5: "Any data fields without accessor methods like this will result in that data being ignored in the process of marshalling." If it is determined that an object-oriented programming language class does not implement a predefined interface including accessor methods, it is ignored.),

said predefined interface comprising annotations including associating said object-oriented programming language class field with a corresponding XML element tag (See McLaughlin-4 page 3 paragraph 2: "Once the element name is in place, you must obtain the attributes. Each attribute should be the **name of the field** and the field's value.")

specifying an object-oriented programming language class to be instantiated when constructing said object-oriented programming language class field from said XML file (See McLaughlin-4 page 3 paragraph 1: "First, the name of the Java class is obtained. This name will become the element name of the XML representation being constructed.")

identifying an object-oriented programming language method to invoke for retrieving said object-oriented programming language class field (See McLaughlin-4 page 2 paragraph 5: "...it is expected to have **accessor methods** for all of its data." Accessor methods are known for retrieving class fields.), *and,*

in the case of said loaded object-oriented programming language class implementing said predefined interface

iteratively processing each field descriptor within the loaded object-oriented programming language class to retrieve corresponding XML tag (See McLaughlin-4 page 2 paragraph 7: "It takes an object and should return an **XML element representation** of that object. If the object contains references to other Java objects, then you can **recurse**..." As noted earlier, if the accessor method is not implemented, the descriptor is ignored.)*; and*

transferring field values to new elements created using said corresponding XML tags (See McLaughlin-4 page 4 paragraph 1: "And once the recursion unwinds, the result is a complete XML representation of the top-level Java object, which is usable as a root **element** in an XML document." Field values are inherently transferred to the resulting XML representation.).

McLaughlin-4 also discloses the use of parameters in defining the marshalling function (See page 3, Listing 2 for the definition of "getXMLRepresentation").

McLaughlin-4 does not expressly disclose defining a method with four parameters, or identifying a JAVA method to invoke for retrieving this method.

However, official notice is taken that the use of parameters in method and interface definitions is known in the art. An arbitrary number of parameters can be used in the definition of a method depending on the scope of the data being operated upon, and the requirements of the method or interface. Parameters are used in method and interface definitions on occasions where proper data values are not available to the called method.

Further, in an analogous environment, McLaughlin-3 teaches using an interface including mutator methods which convert data stored in an XML representation of a Java class instance, into a new Java instance of the same respective class (See page 3 paragraph 2: "Once a new instance of the appropriate class is created, the mutator methods (all named setXXX) are called with the values supplied in the XML document." According to McLaughlin's interface, the Java method to invoke for retrieving this method is standardized based on the name of the attribute.). The mutator method is inherently identified for retrieval purposes.

It would have been obvious to one of ordinary skill in the art at the time the invention was made to use the marshalling method of McLaughlin-4 with the derived mutators of McLaughlin-3 and additional parameters. One of ordinary skill would have been motivated to provide two-way conversions not only from Java to XML, but also from XML to Java by using mutators to initialize data members in an object instance. One would have also been motivated to use parameters to pass data to methods which would otherwise be out of scope.

As per claim 3, McLaughlin-4 discloses:

annotating said object-oriented programming language object to be converted to said XML to include the steps of:

identifying a respective XML tag (See page 3 paragraph 2: "Once the element name is in place, you must obtain the attributes. Each attribute should be the **name of the field** and the field's value." The name of Java attributes are inherently annotated within the Java object.)*;*

identifying an object-oriented programming language class to be instantiated when constructing said object-oriented programming language object field from an XML file, (See page 3 paragraph 1: "First, the name of the Java class is obtained. This name will become the element name of the XML representation being constructed.")*;* and

identifying an object-oriented programming language method to invoke for retrieving said object-oriented programming language object (See page 2 paragraph 5:

"...it is expected to have accessor methods for all of its data." Accessor methods are known for retrieving objects.).

McLaughlin-4 does not expressly disclose *identifying an object-oriented programming language method to invoke for retrieving said retrieval method.*

However, in an analogous environment, McLaughlin-3 teaches using an interface including mutator methods which convert data stored in an XML representation of a Java class instance, into a new Java instance of the same respective class (See page 3 paragraph 2: "Once a new instance of the appropriate class is created, the mutator methods (all named setXXX) are called with the values supplied in the XML document." According to McLaughlin's interface, the Java method to invoke for retrieving this method is standardized based on the name of the attribute.).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to use the marshalling method of McLaughlin-4 with the derived mutators of McLaughlin-3. One of ordinary skill would have been motivated to provide for future unmarshalling of a marshaled object.

As per claim 4, McLaughlin-4 discloses:

retrieving a field description; determining object-oriented programming language conversion parameters

by examining an annotation associated with each object-oriented programming language element to be converted to XML (See McLaughlin-4 page 2 paragraph 7: "It takes an **object** and should return an XML element **representation** of that object. If the object

contains references to other Java objects, then you can recurse..." The examination of annotations is inherent when processing objects.),

said annotation defining for each object-oriented programming language element at least a corresponding XML tag, a corresponding object class (See McLaughlin-4 page 3 paragraph 1: "First, the name of the Java **class** is obtained. This name will become the **element name** of the XML representation being constructed."), *and*

a corresponding field retrieval method (See McLaughlin-4 page 2 paragraph 5: "...it is expected to have **accessor methods** for all of its data." Accessor methods are known for retrieving objects fields.).

McLaughlin-4 does not expressly disclose *a corresponding method retrieval method.*

However, in an analogous environment, McLaughlin-3 teaches using an interface including mutator methods which convert data stored in an XML representation of a Java class instance, into a new Java instance of the same respective class (See page 3 paragraph 2: "Once a new instance of the appropriate class is created, the mutator methods (all named setXXX) are called with the values supplied in the XML document." According to McLaughlin's interface, the Java method to invoke for retrieving this method is standardized based on the name of the attribute.). The mutator method is inherently identified for retrieval purposes.

It would have been obvious to one of ordinary skill in the art at the time the invention was made to use the marshalling method of McLaughlin-4 with the derived mutators of McLaughlin-3. One of ordinary skill would have been motivated to provide

two-way conversions not only from Java to XML, but also from XML to Java by using mutators to retrieve methods to initialize data members in an object instance.

As per claim 5, the rejection of claim 4 is incorporated, and McLaughlin further discloses:

loading a named object-oriented programming language class (See McLaughlin-4 page 2 paragraph 7: "It takes an object and should return an XML element representation of that object." An object is inherently loaded from a JAVA class.)*;*

determining if a loaded object-oriented programming language class implements a predefined interface (See McLaughlin-4 page 2 paragraph 5: "Any data fields without accessor methods like this will result in that data being ignored in the process of marshalling." If it is determined that a JAVA class does not implement accessor methods, it is ignored.),

in the case of said loaded object-oriented programming language class implementing said predefined interface

iteratively processing each field descriptor within the loaded object-oriented programming language class to retrieve the

corresponding XML tag (See McLaughlin-4 page 2 paragraph 7: "It takes an object and should return an XML element representation of that object. If the object contains references to other Java objects, then you can recurse..." As noted earlier, if the accessor method is not implemented, the descriptor is ignored.)*; and*

transferring field values to new elements created using said corresponding XML

tags (See McLaughlin-4 page 4 paragraph 1: "And once the recursion unwinds, the result is a complete XML representation of the top-level Java object, which is usable as a root element in an XML document.").

McLaughlan-4 does not expressly disclose defining a method with four parameters.

However, official notice is taken that the use of parameters in method and interface definitions is known in the art. An arbitrary number of parameters can be used in the definition of a method depending on the scope of the data being operated upon, and the requirements of the method or interface. Parameters are used in method and interface definitions on occasions where proper data values are not available to the called method.

It would have been obvious to one of ordinary skill in the art at the time the invention was made to use the marshalling method of McLaughlin-4 with additional parameters. One of ordinary skill would be motivated to provide the marshalling method with any data that a programmer would want to persist.

As per claim 6, the rejection of claim 4 is incorporated, and McLaughlin-4 further discloses:

instantiating an object of a desired object-oriented programming language class (See McLaughlin-4 page 2 paragraph 7: "It takes an object and should return an XML element representation of that object." An object is inherently instantiated from a JAVA class);

in the case of said instantiated object implementing a predefined interface (See McLaughlin-4 page 2 paragraph 5: "Any data fields without accessor methods like this will result in that data being ignored..."),

iteratively processing each object included within said instantiated object (See McLaughlin-4 page 2 paragraph 7: "If the object contains references to other Java objects, then you can recurse...")

according to the steps of:

retrieving field descriptors associated with an object being processed (See McLaughlin-4 page 3 paragraph 2: "Once the element name is in place, you must obtain the attributes. Each attribute should be the name of the field and the field's value.");

McLaughlin-4 does not expressly disclose creating a specified Java object for each XML element, and storing that object.

However, McLaughlin-3 teaches: *creating an object of specified object-oriented programming language type for each XML element corresponding to a field descriptor* (See McLaughlin-3 page 3 paragraph 2: "Once a new instance of the appropriate class is created, the mutator methods are called with the values supplied in the XML document."; also page 4 paragraph 4: "Once all the attributes are read and assigned to the created Java instance, you need to take each nested element and perform the unmarshalling again.");

and

storing the created object in the currently processed object (See McLaughlin-3 page 4 paragraph 4: "Once all the attributes are read and assigned to the created Java instance).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to use the object processing of McLaughlin-4 with the XML translation of McLaughlin-3. One of ordinary skill would have been motivated store an accurate and complete representation of an object that could easily be sent through a network and reconstituted at a later time.

As per claim 7, McLaughlin-4 discloses:

associating said object-oriented programming language class field with a corresponding XML element tag (See McLaughlin-4 page 3 paragraph 1: "First, the name of the Java class is obtained. This name will become the element name of the XML representation being constructed.");

specifying an object-oriented programming language class to be instantiated when constructing said object-oriented programming language class field from said XML file (See McLaughlin-4 page 3 paragraph 2: "Once the element name is in place, you must obtain the attributes. Each attribute should be the name of the field and the field's value.");

identifying an object-oriented programming language method to invoke for retrieving said object-oriented programming language class field (See McLaughlin-4 page 2 paragraph 5: "...it is expected to have **accessor methods** for all of its data." Accessor methods are known for retrieving class fields.).

McLaughlin-4 also discloses the use of parameters in defining the marshalling function (See page 3, Listing 2 for the definition of "getXMLRepresentation").

McLaughlin-4 does not expressly disclose the use of four parameters, or *retrieving the object-oriented programming language class field.*

However, official notice is taken that the use of parameters in method and interface definitions is known in the art. An arbitrary number of parameters can be used in the definition of a method depending on the scope of the data being operated upon, and the requirements of the method or interface. Parameters are used in method and interface definitions on occasions where proper data values are not available to the called method.

Further, in an analogous environment, McLaughlin-3 teaches using an interface including mutator methods which convert data stored in an XML representation of a Java class instance, into a new Java instance of the same respective class (See page 3 paragraph 2: "Once a new instance of the appropriate class is created, the mutator methods (all named setXXX) are called with the values supplied in the XML document." According to McLaughlin's interface, the Java method to invoke for retrieving this method is standardized based on the name of the attribute. Thus the ability to retrieve a field of a class is provided by the mutator methods.).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to use the marshalling method of McLaughlin-4 with the mutator methods of McLaughlin-3 along with multiple parameters. One of ordinary skill would have been motivated to provide two-way conversions not only from Java to XML, but also from XML to Java by using mutators to initialize data members in an object instance. One would have also been motivated to use parameters to pass data to methods which would otherwise be out of scope.

As per claim 8, the rejection of claim 7 is incorporated.

McLaughlin-4 discloses the XML representation of Java collections (See page 2 paragraph 7).

McLaughlin-4 does not expressly disclose *specifying a type of object-oriented programming language object to instantiate for an XML element representing a collection.*

However, in an analogous environment, McLaughlin-3 teaches conversion of a supplied string value in XML to a Java type (See Listing 4, also page 4 paragraph 2: "Once this data has been converted to the appropriate type, reflection can be used to invoke the accessor method and pass in the converted data type. This enables all attributes and their values in the XML document to be stored as method variables and values in the resulting Java instance.")

It would have been obvious to one of ordinary skill in the art at the time the invention was made to use the marshalling method of McLaughlin-4, with the type specification of McLaughlin-3. One of ordinary skill would have been motivated to supply type information to an XML representation of a Java class. Strong typing is essential to the Java programming language.

All further limitations have been addressed in the above rejection of claim 7.

As per claim 9, the rejection of claim 8 is incorporated.

McLaughlin-4 further discloses *specifying a tag name to use for each element representing a collection* (See McLaughlin-4 page 3 paragraph 1: "First, the name of the Java class is obtained. This name will become the element name of the XML representation being constructed.").

All further limitations have been addressed in the above rejection of claim 7.

8. Claim 10 is rejected under 35 U.S.C. 103(a) as being unpatentable over McLaughlin-4 in view of McLaughlin-3 as applied to claim 8 above, and further in view of "Data Binding from XML to Object-oriented programming language Code, Part2" by McLaughlin published in August 2000 (hereinafter referred to as "McLaughlin-2").

The text of the rejection of claim 10 below is reproduced for convenience from the previous Office action filed 19 May 2004 in light of the amendment filed 10 September 2004.

McLaughlin-4 does not expressly disclose the use of a hash table.

However, in an analogous environment, McLaughlin-2 teaches the use of a hash table in Java to store multiple interfaces and implementations (See page 2 paragraph 5)

It would have been obvious to one of ordinary skill in the art at the time the invention was made to use the collection of McLaughlin-4 with the hash table of McLaughlin-2. One of ordinary skill would have been motivated to represent Java objects and data in a compact and easily searchable form.

9. Claims 11-13 are rejected under 35 U.S.C. 103(a) as being unpatentable over McLaughlin-4 in view of McLaughlin-3 further in view of U.S. Patent 5,367,685 to Gosling (hereinafter referred to as "Gosling").

The text of the rejections of claims 11-13 below is reproduced for convenience from the previous Office action filed 19 May 2004 in light of the amendment filed 10 September 2004.

As per claim 11, the combination of McLaughlin-4 and McLaughlin-3 does not expressly disclose a computer system comprising processing means for executing and memory means for storing.

However, in an analogous environment, Gosling teaches the use of a computer system comprising processing means for executing and memory means for storing (Figure 2, elements 22 and 24; also column 3 lines 55-57: "As shown in FIG. 2, the exemplary computer system 20 comprises a central processing unit (CPU) 22, a memory 24, and an I/O module 26.").

All further limitations have been addressed in the above rejection of claim 4.

It would have been obvious to one of ordinary skill in the art at the time the invention was made to use Gosling's computer system to execute McLaughlin-4's marshalling method. One of ordinary skill would have been motivated to execute McLaughlin-4's system on a computer system in order to produce a tangible result.

As per claims 12 and 13, the above rejection of claim 11 is incorporated. All further limitations have been addressed in the above rejections of claims 5 and 6, respectively.

Conclusion

10. This is a continuation of applicant's earlier Application No. 09/837,929. All claims are drawn to the same invention claimed in the earlier application and could have been finally rejected on the grounds and art of record in the next Office action if they had been entered in the earlier application. Accordingly, **THIS ACTION IS MADE FINAL** even though it is a first action in this case. See MPEP § 706.07(b). Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no, however, event will the statutory period for reply expire later than SIX MONTHS from the mailing date of this final action.

11. Any inquiry concerning this communication or earlier communications from the examiner should be directed to J. Derek Rutten whose telephone number is (571) 272-3703. The examiner can normally be reached on T-F 6:00 - 4:30.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Tuan Q. Dam can be reached on (571) 272-3695. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

jdr

TUAN DAM
SUPERVISORY PATENT EXAMINER